

06-23-00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventorship..... Ruthfield et al.
 Applicant..... Microsoft Corporation
 Attorney's Docket No. MS1-557US
 Title: Methods and Systems of Providing Information to Computer Users

1c844 U.S. PTO
 09/599806

06/21/00

TRANSMITTAL LETTER AND CERTIFICATE OF MAILING

To: Commissioner of Patents and Trademarks,
 Washington, D.C. 20231

From: Lance R. Sadler (Tel. 509-324-9256; Fax 509-323-8979)
 Lee & Hayes, PLLC
 421 W. Riverside Avenue, Suite 500
 Spokane, WA 99201

The following enumerated items accompany this transmittal letter and are being submitted for the matter identified in the above caption.

1. Specification—title page, plus 56 pages, including 84 claims and Abstract
2. Transmittal letter including Certificate of Express Mailing
3. 11 Sheets Formal Drawings (Figs. 1-14)
4. Return Post Card

Large Entity Status ☒ [x]

Small Entity Status ☐ []

Date: 6/21/00

By: 

Lance R. Sadler
 Reg. No. 38,605

CERTIFICATE OF MAILING

I hereby certify that the items listed above as enclosed are being deposited with the U.S. Postal Service as either first class mail, or Express Mail if the blank for Express Mail No. is completed below, in an envelope addressed to The Commissioner of Patents and Trademarks, Washington, D.C. 20231, on the below-indicated date. Any Express Mail No. has also been marked on the listed items.

Express Mail No. (if applicable) EL624352025

Date: 6/21/2000

By: 

Lori A. Vierra

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

**Methods and Systems of Providing Information to
Computer Users**

Inventor(s):

Scott Ruthfield

Richard Banks

ATTORNEY'S DOCKET NO. MS1-557US

RELATED APPLICATIONS

The following patent applications are related to the present application, are assigned to the assignee of this patent application, and are expressly incorporated by reference herein:

- U.S. Patent Application Serial No. _____, entitled "Single Window Navigation Methods and Systems", bearing attorney docket number MS1-560us, and filed on the same date as this patent application;
- U.S. Patent Application Serial No. _____, entitled "Methods, Systems, Architectures and Data Structures For Delivering Software via a Network", bearing attorney docket number MS1-559us, and filed on the same date as this patent application;
- U.S. Patent Application Serial No. _____, entitled "Network-based Software Extensions", bearing attorney docket number MS1-563us, and filed on the same date as this patent application;
- U.S. Patent Application Serial No. _____, entitled "Authoring Arbitrary XML Documents using DHTML and XSLT", bearing attorney docket number MS1-583us, and filed on the same date as this patent application;
- U.S. Patent Application Serial No. _____, entitled "Architectures For And Methods Of Providing Network-based Software Extensions", bearing attorney docket number MS1-586us, and filed on the same date as this patent application.
- U.S. Patent Application Serial No. _____, entitled "Task Sensitive Methods And Systems For Displaying Command Sets", bearing attorney docket number MS1-562us, and filed on the same date as this patent application.

TECHNICAL FIELD

This invention pertains to computerized methods and systems for providing information to computer users. More particularly, the invention concerns methods and systems for providing links to user-specific information.

BACKGROUND

As computing evolves, there is a growing demand to make the user's computing experience much more user-centric, or tailored to the particular user. Consider, for example, web browsers. Web browsers are application programs that execute on a user's computer and enable a user to navigate the web and search for content, typically in the form of web pages that are displayed on their computer. To make the user's web browsing experience more user-centric, most web browsers include what is known as a "add favorite" function in which a user can add particular web sites to a "favorites" list. When a user adds a web site or web page to their favorites list (in the form of a link), they physically create an entry in a database that maintains a URL (i.e. universal resource locator) associated with the web site or page. Anytime a user wishes to return to one of these web sites or pages, they simply pull up their "favorites" list, click on the appropriate link, and their web browser obtains and displays a web page that corresponds to the link.

Another way that web browsers attempt to create a user-centric experience is by keeping track of a very limited amount of so-called "history data" pertaining to the user's historical browsing activities. History data might include the last three web sites that were browsed by the user. The user can typically view this information by clicking on a feature that provides a drop down menu that lists links to the browsed sites. For example, on the web browser's navigation bar, there is typically a "back" and "forward" button that can be clicked by the user to navigate backward or forward among entries that are maintained in a navigation stack that keeps track of the user's browsing activities. The "back" and "forward" buttons can also have drop down menus associated with them that enable the user

1 to display a drop down menu that might include links for the last three sites that
2 the user encountered. By selecting one of these links, the user's browser displays
3 the corresponding web page.

4 While these solutions provide a very basic user-centric functionality, they
5 fall far short of providing a versatile, intelligently flexible and dynamic system.
6 For example, many of these systems require the user to initiate or take some action
7 in order for particular links to appear (i.e. the user necessarily must add a link to
8 their favorites list). In addition, many of these systems are unintelligent in the way
9 that they present information or links to the user. For example, a favorites list may
10 have a large number of links that have been added by the user. When a user
11 attempts to find a link to a favorite web site, their browser will typically present
12 them with all of the links that are in their favorites list. It is then up to the user to
13 find the appropriate link so that they can select it.

14 Another challenge in the general area of information use is that which is
15 posed by the move toward context-aware computing systems. Context-aware
16 computing systems are those systems that provide services to a user based upon
17 their context. In the future, information processing systems are going to have to
18 be sensitive to the user's desire to accomplish tasks in context-aware systems. For
19 example, it may be desirable to provide services to a user without requiring the
20 user to change their context in order to consume the services. As an example,
21 consider the following scenario. A user is working in a word processing
22 application on a particular document of interest. The document is provided by an
23 application program that is executing on the user's computer and that displays the
24 document in a window that is defined by the program. Consider now that the user
25 receives four or five email messages during the course of working on the

document. In order to view indicia of these email messages (i.e. the "From" and "Subject" fields), in today's computing environment, the user is typically required to pull up their email application program which separately displays a different window that includes the indicia that the user wishes to view. This is a "modal" operation in that the user is required to temporarily quit working on their document in the word processing application program so that they can view information provided by the email application program. Thus, the user is undesirably required to change their context.

This invention arose out of concerns associated with improving methods and systems that provide information to computer users.

SUMMARY

Methods and systems of providing information to computer users are described. In one embodiment, a user interface is provided having a display area that permits a user to accomplish various tasks. Individual tasks can be associated with individual different functionalities which can enable the user to accomplish tasks in different user contexts, e.g. word processing tasks, email tasks, web browsing tasks and the like. The different tasks can advantageously pertain to different content types or document types. When a user is working within a particular context, they can view quick links that are associated with one or more of the different contexts without having to change their particular context. By clicking on a quick link, the user then can automatically navigate to the context associated with the quick link so that they can then accomplish context-specific tasks. Advantageously, the various functionalities or contexts can be provided by

1 a single application program that also manages the user's navigation activities to
2 and between the different functionalities.

3 In various embodiments, a user can select from among a number of
4 different algorithms that affect which quick links are displayed. The algorithms
5 can advantageously deploy across different content types. The different
6 algorithms include a "Top Favorites" algorithm that presents quick links based on
7 items on their favorites list visited most often by a user in combination with items
8 that have been recently added by a user to their favorites list; a "Suggested
9 Favorites" algorithm that presents quick links based on items visited most often by
10 a user in combination with items that have been recently visited by a user; and a
11 "Recent Items List" that presents quick links to various different documents of
12 different content types that were last visited by the user.

13 14 **BRIEF DESCRIPTION OF THE DRAWINGS**

15 Fig. 1 is a block diagram of an exemplary computer system that can be used
16 to implement various described embodiments.

17 Fig. 2 is a diagram of an exemplary user interface that can be provided in
18 accordance with one described embodiment.

19 Fig. 3 is a flow diagram that describes steps in a method in accordance with
20 one described embodiment.

21 Fig. 4 is a diagram of an exemplary user interface in accordance with one
22 specific implementation.

23 Fig. 5 is a diagram of an exemplary user interface in accordance with one
24 specific implementation.
25

1 Fig. 6 is a diagram of an exemplary user interface in accordance with one
2 specific implementation.

3 Fig. 7 is a flow diagram that describes steps in a method in accordance with
4 one described embodiment.

5 Fig. 8 is a diagram of an exemplary user favorites interface.

6 Fig. 9 is a flow diagram that describes steps in a method in accordance with
7 one described embodiment.

8 Fig. 10 is a diagram that describes a portion of a database in accordance
9 with one described embodiment.

10 Fig. 11 is a flow diagram that describes steps in a method in accordance
11 with one described embodiment.

12 Fig. 12 is a flow diagram that describes steps in a method in accordance
13 with one described embodiment.

14 Fig. 13 is a diagram that illustrates the concept of a "Recent Items List."

15 Fig. 14 is a diagram that illustrates an exemplary implementation of the
16 "Recent Items List."

17 18 **DETAILED DESCRIPTION**

19 **Overview**

20 In various embodiments described just below, novel methods and systems
21 provide so-called browsable "quick links" to user-related data. The quick links
22 can be advantageously deployed in a manner in which the user can browse the
23 quick links without having to change or modify their current computing context.
24 The quick links can be provided across multiple different content types, e.g.
25 document types. Thus, a user can, in some instances, view quick links associated

1 with different content types without having to change their current computing
2 context, i.e. without having to change a document of a particular content type in
3 which they happen to be working.

4 In one particularly advantageous embodiment, multiple different
5 functionalities can be provided by a single application program. The multiple
6 different functionalities enable a user to accomplish multiple different tasks within
7 the context of a single application program. This single application program
8 might, for example, provide multiple document-centric functionalities, e.g. an
9 email functionality, word processing functionality, and web browser functionality.
10 In this example, a user working within the web browser functionality can view
11 quick links associated with the email functionality without having to change their
12 web browsing context. A user is then able to select a link to automatically
13 navigate to a particular document that is associated with that link.

14 Another aspect of some of the described embodiments includes an ability to
15 build the quick links using dynamically-changing information that is not
16 necessarily information that is demanded by the user. That is, in many systems,
17 information will be received that pertains to a particular user. For example, in a
18 single application program that includes an email functionality, a user may, over
19 the course of browsing web sites, receive one or more email messages. These
20 email messages constitute dynamically-changing information which, in this
21 example, is not related to any actions that the user is taking. Nonetheless, quick
22 links to the email messages can be advantageously displayed for the user while
23 they are in the context of their web browsing activities.

24 Other embodiments provide intelligent browsing algorithms that are
25 directed to displaying quick links that are very likely to be of interest to a user.

1 These intelligent browsing algorithms can be advantageously deployed in
2 connection with multiple content-type systems so that the algorithms are adaptable
3 to and address the different content types.

4 Thus, the described embodiments provide very powerful methods and
5 systems that greatly enhance the user's computing experience by, among other
6 things, specifically tailoring the user's computing experience to their particular
7 context. Flexibility is enhanced by providing, in some instances, systems that are
8 configured to work within a context-sensitive computing environment that
9 contains multiple different functionalities that are selectable for use by a user.

10 11 Exemplary Computer System

12 Fig. 1 shows an exemplary computer system that can be used to implement
13 the embodiments described herein. Other computer systems can, however, be
14 used. Computer 130 includes one or more processors or processing units 132, a
15 system memory 134, and a bus 136 that couples various system components
16 including the system memory 134 to processors 132. The bus 136 represents one
17 or more of any of several types of bus structures, including a memory bus or
18 memory controller, a peripheral bus, an accelerated graphics port, and a processor
19 or local bus using any of a variety of bus architectures. The system memory 134
20 includes read only memory (ROM) 138 and random access memory (RAM) 140.
21 A basic input/output system (BIOS) 142, containing the basic routines that help to
22 transfer information between elements within computer 130, such as during start-
23 up, is stored in ROM 138.

24 Computer 130 further includes a hard disk drive 144 for reading from and
25 writing to a hard disk (not shown), a magnetic disk drive 146 for reading from and

1 writing to a removable magnetic disk 148, and an optical disk drive 150 for
2 reading from or writing to a removable optical disk 152 such as a CD ROM or
3 other optical media. The hard disk drive 144, magnetic disk drive 146, and optical
4 disk drive 150 are connected to the bus 136 by an SCSI interface 154 or some
5 other appropriate interface. The drives and their associated computer-readable
6 media provide nonvolatile storage of computer-readable instructions, data
7 structures, program modules and other data for computer 130. Although the
8 exemplary environment described herein employs a hard disk, a removable
9 magnetic disk 148 and a removable optical disk 152, it should be appreciated by
10 those skilled in the art that other types of computer-readable media which can
11 store data that is accessible by a computer, such as magnetic cassettes, flash
12 memory cards, digital video disks, random access memories (RAMs), read only
13 memories (ROMs), and the like, may also be used in the exemplary operating
14 environment.

15 A number of program modules may be stored on the hard disk 144,
16 magnetic disk 148, optical disk 152, ROM 138, or RAM 140, including an
17 operating system 158, one or more application programs 160, other program
18 modules 162, and program data 164. A user may enter commands and
19 information into computer 130 through input devices such as a keyboard 166 and a
20 pointing device 168. Other input devices (not shown) may include a microphone,
21 joystick, game pad, satellite dish, scanner, or the like. These and other input
22 devices are connected to the processing unit 132 through an interface 170 that is
23 coupled to the bus 136. A monitor 172 or other type of display device is also
24 connected to the bus 136 via an interface, such as a video adapter 174. In addition
25

1 to the monitor, personal computers typically include other peripheral output
2 devices (not shown) such as speakers and printers.

3 Computer 130 commonly operates in a networked environment using
4 logical connections to one or more remote computers, such as a remote computer
5 176. The remote computer 176 may be another personal computer, a server, a
6 router, a network PC, a peer device or other common network node, and typically
7 includes many or all of the elements described above relative to computer 130,
8 although only a memory storage device 178 has been illustrated in Fig. 1. The
9 logical connections depicted in Fig. 1 include a local area network (LAN) 180 and
10 a wide area network (WAN) 182. Such networking environments are
11 commonplace in offices, enterprise-wide computer networks, intranets, and the
12 Internet.

13 When used in a LAN networking environment, computer 130 is connected
14 to the local network 180 through a network interface or adapter 184. When used
15 in a WAN networking environment, computer 130 typically includes a modem 186
16 or other means for establishing communications over the wide area network 182,
17 such as the Internet. The modem 186, which may be internal or external, is
18 connected to the bus 136 via a serial port interface 156. In a networked
19 environment, program modules depicted relative to the personal computer 130, or
20 portions thereof, may be stored in the remote memory storage device. It will be
21 appreciated that the network connections shown are exemplary and other means of
22 establishing a communications link between the computers may be used.

23 Generally, the data processors of computer 130 are programmed by means
24 of instructions stored at different times in the various computer-readable storage
25 media of the computer. Programs and operating systems are typically distributed,

1 for example, on floppy disks or CD-ROMs. From there, they are installed or
2 loaded into the secondary memory of a computer. At execution, they are loaded at
3 least partially into the computer's primary electronic memory. The invention
4 described herein includes these and other various types of computer-readable
5 storage media when such media contain instructions or programs for implementing
6 the steps described below in conjunction with a microprocessor or other data
7 processor. The invention also includes the computer itself when programmed
8 according to the methods and techniques described below.

9 For purposes of illustration, programs and other executable program
10 components such as the operating system are illustrated herein as discrete blocks,
11 although it is recognized that such programs and components reside at various
12 times in different storage components of the computer, and are executed by the
13 data processor(s) of the computer.

14 15 **Viewable Links Across Multiple Content Types**

16 In one embodiment, software provides various quick links that are viewable
17 by a user without changing their current computing context. The quick links can
18 advantageously pertain to another context that is different from a context in which
19 the user is currently computing. For purposes of this document, a "context" can be
20 considered as a functionality that enables a user to complete a particular
21 computing task. Exemplary contexts include, without limitation, word processing
22 contexts, browsing contexts, email contexts and the like. Thus, while a user is in
23 one particular context, they have the ability to view quick links that pertain to
24 another different context without changing their current computing context.

As an example, consider the following. A user is currently working on a document in a word processor. During the course of working on the document, the user receives an email message from a friend. In the past, the user would have to temporarily stop their current computing context, e.g. by exiting or pausing a word processing window, and pulling up an email window to view indicia of the email message (i.e. the "From" and "Subject" fields). In accordance with this example, a user can view quick link that are associated with the different context without changing their current context. In this specific case, the user can view links that are associated with the email message (e.g. the "From" and "Subject" fields) without exiting the word processing window or changing their word processing context.

Fig. 2 shows but one exemplary user interface 200 that can be provided in accordance with this example. The user interface is implemented in software that is executable on a user's computing device, e.g. a personal computer, although any computing device can suffice. Interface 200 includes, in this example, a tool bar 202 and a display area 204. Display area 204 can be used by the user to engage in tasks associated with a first context and, in this example, is designated "First Contextual Display." Exemplary tasks can be any suitable tasks in which a computer can engage. Exemplary non-limiting tasks were mentioned above. Tool bar 202 includes, in this example, user-engagable indicia 206 that can enable a user to view quick links that are associated with one or more contexts that are different from the first context and each other. In this example, the indicia comprises one or more drop down menus 206. Each drop down menu can be associated with a different context, i.e. different task, in which a user can engage.

1 In this particular example, and for the sake of brevity, only one indicia or drop
2 down menu is shown.

3 In operation, a user who is working within a particular context in display
4 area 204 may desire to view links associated with a different context. In this case,
5 the user simply clicks on the drop down menu 206 to automatically view one or
6 more quick links that are associated with a different context. When the user clicks
7 on the menu 206, their context within the display area does not change. That is,
8 they are able to view the quick link or links associated with the different context or
9 contexts, without having to change their own context. To this extent, the display
10 of the quick links associated with the other context is done in a modeless fashion.
11 That is, when the user displays the quick links, they are not required to temporarily
12 leave their current context. They may continue working within or at least view
13 their current context in the display area 204 while the quick links are displayed.

14 As an example, consider again the user who is working in a word processor
15 on a particular document and receives an email message from a friend. Instead of
16 having to leave the current document displayed in display area 204, the user
17 simply clicks on the menu 206 to view of list of quick links that correspond to the
18 email messages that the user has received. In this way, the user can check their list
19 of email messages (or view links that pertain to one context) while working in a
20 completely different context. The user can then click on a quick link to be
21 navigated to the new context which, in this case, is the email message.

22 Fig. 3 is a flow diagram that describes steps in a method in accordance with
23 the described embodiment. The illustrated method can be implemented in any
24 suitable hardware, software, firmware, or combination thereof. In the illustrated
25 example, the method is implemented in software.

Step 300 receives information that pertains to different user contexts. This information can comprise any information that can be associated with any number of different user contexts. Advantageously, the information can comprise dynamically changing information. For example, the information can comprise “incoming” information that is received by the user’s computer while the user is working within one context (e.g. receiving an email message while working in a word processing document). Such information can also comprise information that is or is not generated by the user themselves. The information can also comprise time-sensitive information (e.g. calendar appointments for a particular day or time frame), in which case the information would appear when the appointments start in the immediate future. Further, the information can comprise information concerning content that the user is working with and information about the content’s use. Step 302 presents a display on the user’s computer that pertains to a first user context. The display can be any suitable display with which the user can interact to accomplish a task. Step 304 presents user-engagable indicia that enables a user to access quick links associated with one or more contexts that are different from the first context. The links can be associated with the information that is received at step 300. Any suitable user-engagable indicia can be presented. In the example given above, the indicia is displayed in the form of a drop down menu. Step 306 displays quick links that are associated with the different contexts responsive to a user engaging the user-engagable indicia. In the above example, this step can be implemented when the user clicks on the drop down menu that is associated with the different contexts. Advantageously, steps 304 and 306 are implemented without changing the user’s present computing context. So, in the above example, this step is implemented by displaying quick links to the user’s

1 incoming email messages without requiring the user to change their word
2 processing context.

3 4 **Exemplary Implementation**

5 In accordance with one specific implementation, software provides a user
6 interface (UI) that presents a user with a single navigable window that can be
7 navigated between multiple different functionalities by a user. The single
8 navigable window and different functionalities are advantageously provided by a
9 single application program which greatly facilitates integration of the different
10 functionalities. The single navigable window contains user-engagable indicia that
11 enables a user to view quick links that are associated with different functionalities,
12 without having to change their current context or functionality. An exemplary
13 single navigable window application is described in the U.S. Patent Application
14 entitled "Single Window Navigation Methods and Systems", incorporated by
15 reference above.

16 In the exemplary single navigable window application, a user, through the
17 use of various navigation instrumentalities, can navigate between the
18 functionalities and when doing so, the single window presents one of these
19 functionalities. When this one functionality is presented to the user, the user is
20 able, through the use of the user-engagable indicia, to view quick links associated
21 with one or more of the other functionalities. In this particular implementation,
22 one navigation instrumentality is provided in the form of a web browser-like
23 navigation tool. The choice of a web browser-like navigation tool follows from
24 concerns that navigation instrumentalities be of a type that is readily understood
25 by most individuals familiar with computing environments. Thus, when a user

1 first encounters the inventive navigable single window concept for the first time,
2 they do not have to learn an unfamiliar navigation concept. Another navigation
3 instrumentality includes links to each of the multiple different functionalities.
4 These links are different from the quick links and can be clicked on by a user to
5 automatically navigate the single navigable window to a selected functionality.
6 Once the user has navigated the single window to a particular functionality, they
7 can set about accomplishing a task within the functionality. One or more of the
8 application links includes the user-engagable indicia that, in turn, displays the
9 quick links to the associated functionality.

10 Fig. 4 shows but one exemplary user interface (UI) 400 in accordance with
11 this specific implementation. It will be appreciated that other UIs could be used to
12 implement the inventive concepts described herein and that the illustrated UI
13 constitutes but one way of doing so. In the illustrated example, UI 400 includes a
14 navigation bar 402, one or more command areas 404, and a display or document
15 area 406 that constitutes the single navigable window.

16 Navigation bar 402 is located adjacent the top of display area 406 and
17 contains browser-like navigation buttons 408 in the form of a “backward” button,
18 a “forward” button, a “stop” button and the like. The navigation bar can be
19 located anywhere on the UI. Its illustrated placement, however, is similar in
20 appearance to the placement of traditional web browsing navigation features. In
21 addition to the navigation buttons 408, the navigation bar 402 also includes one or
22 more links 410 to the different functionalities that are provided by the single
23 application program and which can be accessed by the user. Individual links 410
24 have user-engagable indicia 411 associated with them that enable a user to view
25 quick links that are associated with the functionality. In the illustrated example,

1 links to three exemplary functionalities (i.e. functionality 1, functionality 2, and
2 functionality 3) are shown and each has its own user-engagable indicia 411. It is
3 possible, however, for less than all of the functionalities to have user-engagable
4 indicia. These functionalities are typically different functionalities that can enable
5 a user to complete different respective tasks. Examples of different tasks are given
6 below in more detail. In this example, these functionalities are advantageously all
7 provided within the context of a single application.

8 In operation, to access a particular functionality, a user simply clicks on one
9 of the links 410 and a display that pertains to the selected functionality is
10 immediately presented in the single window display area 406. To view quick links
11 that are associated with a particular functionality that is the same as or different
12 from one in which the user is currently working, the user simply clicks on the
13 corresponding user-engagable indicia 411 to see a drop down menu containing the
14 quick links. Thus, while working within functionality 1, for example, the user
15 could click on the user-engagable indicia 411 associated with any of
16 functionalities 1, 2 and 3 to see their associated quick links. By clicking further on
17 any of the quick links, the user can automatically navigate the single window to
18 that particular link. As the user navigates from link to link or from functionality to
19 functionality, their navigation activities are managed by a software-implemented
20 navigation model that is described in a section entitled "Navigation Model" below.

21 Command areas 404 are located adjacent the top and left side of the display
22 area 406. The command area(s) can, however, be located in any suitable location.
23 The command areas provide commands that are both global in nature and specific
24 to the particular context the user has selected. For example, some commands such
25 as "search" and "help" might be considered as global in nature since they can find

1 use in many contexts. Other commands, such as “text bold” or “forward” are
2 more specific to the particular context that the user has selected. For the “text
3 bold” command, the user’s context may likely be a word processing context, while
4 the “forward” command may likely be employed in an email context. The concept
5 of context-sensitive command structures are described in more detail in the U.S.
6 Patent Application entitled “Task Sensitive Methods And Systems For Displaying
7 Command Sets”, incorporated by reference above.

8 Briefly, however, context-sensitive command structures include command
9 sets having one or more individual commands are automatically presented to a
10 user depending on the user’s context. Specifically, depending on the type of
11 action the user has taken within display area 406, commands that are specific to
12 that action will appear automatically thus obviating the need for the user to hunt
13 through a menu structure to find commands of interest. This improves upon past
14 approaches which always presented top level commands, even when they were not
15 needed by the user. This is also advantageous from the standpoint of assisting
16 users who are unfamiliar with a particular software application. In the past, these
17 users would have to hunt through an unfamiliar menu structure to find commands
18 that may or may not be pertinent to an action that the user desired to take. In the
19 present case, contextually-appropriate commands are automatically presented in
20 an interface so that a user need not worry about finding appropriate commands.

21 In the present example, a context-sensitive command structure in the form
22 of a context block can be presented to the user. The context block can
23 advantageously contain multiple algorithms from which the user can select to see
24 different collections of links that pertain to the particular functionality in which
25 they are currently working or one or more of the functionalities in which they are

not currently working. The algorithms are designed to intelligently present links that are very likely to be of interest to the user. Exemplary algorithms are described in more detail in the “Exemplary Algorithms” section below.

Example

As an example of the single navigable window provided by a single application consider Figs. 5 and 6.

In this example, the multiple functionalities 410 that can be navigated by a user include a browser functionality (indicated by the home icon), a mail functionality (indicated by the letter icon), a planner functionality (indicated by the clock icon), a contacts functionality (indicated by the people icon), a documents functionality (indicated by the folder icon), and a links functionality (indicated by the world icon). These illustrated functionalities are so-called “document-centric” functionalities because they are defined around a type of document that a user interacts with, e.g. a Web page document, an email document, a calendar document, etc. Each of the links 410 to the functionalities has an associated user-engagable indicia 411 in the form of a drop down menu that shows quick links to the various functionalities.

Fig. 5 shows an example of a display that is rendered in the display area 406 when a user clicks on the link to the browser functionality. By clicking on the link (i.e. the home icon) to the browser functionality, single application program software executing on the user’s computer executes to implement a browser functionality. In this example, the browser functionality displays the user’s home page in display area 406. Notice also that navigation buttons 408 are provided for navigation within the current and between the different selectable functionalities.

1 The command areas 404 contain command sets that include commands that are
2 specific to the context that the user has selected. In this example, the user's
3 context is a browsing context. Accordingly, the leftmost command area contains
4 commands that are specific to the browsing functionality. Such commands
5 include ones that a user would normally expect to find in a web browser. In
6 addition, the leftmost command area 404 shows a context block 412 labeled
7 "Favorites" that includes a drop down menu that can enable a user to select
8 between multiple different algorithms that intelligently present links associated
9 with their current context. In this example, context block 412 indicates that the
10 user has selected a "Top Favorites" algorithm that lists their top favorite web sites.
11 An exemplary "Top Favorites" algorithm is described below in the "Exemplary
12 Algorithms" section.

13 Notice also that the command area 404 adjacent the top of display area 406
14 also contains commands that are specific to the browsing context, i.e. "Add to
15 Favorites" and an address well in which the user can type a URL of a particular
16 destination web site.

17 Fig. 6 shows an example of a display that is rendered in the display area
18 406 when the user clicks on the link to the mail functionality (i.e. the folder icon).
19 By clicking on this link, single application program software executing on the
20 user's computer executes to implement the mail functionality. In this example, the
21 mail functionality displays a user's in box with messages that have been received
22 by the user. Notice that the leftmost command area has been minimized by the
23 user and that the command area adjacent the top of the display area 406 contains
24 commands that are specific to the user's current context, e.g. "New" for generating
25

1 a new email message, "Reply" for replying to an email message, "Reply to All"
2 for replying to all recipients of an email message and the like.

3 Likewise, although not specifically illustrated, the user could have displays
4 for the planner, contacts, documents, and links functionalities presented in the
5 display area 406 by simply clicking on the links to these specific functionalities.
6 The navigation bar 408 provides the user with the ability to navigate through these
7 different functionalities in a browser-like manner.

8 It is important to note that the above example constitutes but one exemplary
9 way in which multiple different functionalities and associated quick links can be
10 presented to a user within the construct of a navigable structure. It should be
11 understood that the specifically illustrated functionalities (i.e. browser, mail,
12 planner etc.) constitute specific examples of different functionalities that are
13 capable of being incorporated into the single application program that provides the
14 navigable window and should in no way limit the scope of the claimed subject
15 matter to only the specifically illustrated and described functionalities.
16 Accordingly, other different functionalities and associated quick links can be
17 employed.

18 Fig. 7 is a flow diagram that describes steps in a method in accordance with
19 this described embodiment. The illustrated method can be implemented in any
20 suitable hardware, software, firmware, or combination thereof. In the illustrated
21 example, the method is implemented in software.

22 Step 700 provides a single application program with multiple different
23 functionalities. The functionalities, as pointed out above, are advantageously
24 different so as to enable a user to accomplish different tasks. One specific non-
25 limiting example of different functionalities was given above in the context of

document-centric functionalities that enable a user to make use of browser, mail, planner, contacts, documents, and links functionalities. Step 700 can be implemented by configuring a computing device, such as a user's computer, with the single application program having the multiple different functionalities. This step can also be implemented by providing a software platform in the form of a generic single application shell that is extensible and adaptable to receive different extensions or software modules that embody various different functionalities as described in the U.S. Patent Applications entitled "Single Window Navigation Methods and Systems", "Methods, Systems, Architectures and Data Structures For Delivering Software via a Network", and "Network-based Software Extensions" incorporated by reference above. These different extensions are then presented to the user in the context of the single application having the multiple different functionalities.

These extensions can be delivered to the platform in any suitable way and through any suitable delivery mechanism. For example, one way of delivering the various extensions or functionalities is to deliver them via a network such as an Intranet or the Internet. Regardless of the manner in which the single application is provided, step 702 presents a user interface (UI) with a single window, links to the multiple different functionalities, and user-engagable indicia associated with one or more of the links. The user-engagable indicia, as described above, enables a user to access quick links associated with one or more of the functionalities. The UI can also advantageously include navigation instrumentalities that enable a user to navigate between the different functionalities in a browser-like manner. Figs. 4-6 give specific examples of an exemplary UI that can be used in accordance with the described embodiment. Step 704 ascertains whether a user has engaged any of

1 the user-engagable indicia for displaying the quick links. If the user has not
2 engaged any of the user-engagable indicia, then step 705 does not display any of
3 the quick links. The user-engagable indicia can be continually displayed so that a
4 user is free to select one. If the user has engaged any of the user-engagable indicia
5 (e.g. by clicking on a drop down menu 411 associated with one or more of the
6 functionalities), then step 706 displays the quick links that are associated with the
7 user-engagable indicia. Step 708 ascertains whether the user has selected a
8 particular quick link from the displayed quick links. If the user has not, then step
9 710 can remove the display of quick links and branches back to step 704. This
10 step can be implemented automatically (e.g. by removing the quick links display
11 after a determinable amount of time) or manually (by enabling the user to close the
12 quick links display through some predefined action). If the user has selected a
13 particular quick link, then step 712 navigates the single window to the selected
14 quick link and displays a document associated with the quick link for the user.
15 Step 712 then returns to step 704. It will be appreciated that step 706 can also
16 remove quick links that are displayed responsive to a user engaging the user-
17 engagable indicia.

18 Hence, in this example, multiple different functionalities are provided by a
19 single application program that provides a single navigable window that can be
20 navigated among the different functionalites. This permits a user to accomplish
21 different tasks without having to pull up and manage multiple windows. All of the
22 functionalities, in this example, are provided within the single window as desired
23 by the user. To assist the user in operating within the single window environment,
24 one or more of the functionalities have user-engagable indicia associated with
25 them that enables a user to view quick links that pertain to a functionality that is

1 different from a functionality in which they happen to be working.
2 Advantageously, the user is able to view the quick links without having to change
3 their current context. For example, in the document-centric example described in
4 Figs. 5 and 6, a user can view quick links associated with upcoming appointments
5 in their calendar functionality while browsing the web with their browser
6 functionality. When they view the links to the appointments, their context remains
7 within the browser functionality. If the user chooses, they may click on a
8 particular quick link to an appointment which then changes their context and
9 navigates the single navigable window to a document that displays more
10 information about the appointment.

11 12 **Navigation model**

13 In the embodiment described directly above, a navigation model is utilized
14 to manage a user's navigation activities within the single application that provides
15 the multiple different functionalities. Although any suitable navigation model (as
16 will be understood by those of skill in the art) can be used, in the described
17 embodiment a so-called "back-and-truncate" navigation stack is used. The basic
18 concept of a back-and-truncate model is known and forms the basis for many
19 different web browsers on the market today. Essentially, the back-and-truncate
20 model makes use of a navigation stack that is truncated when the user navigates
21 back n times and then forward to a new document. An explanation of the
22 navigation model that is employed in the present example is given in the U.S.
23 Patent Application entitled "Single Window Navigation Methods and Systems",
24 incorporated by reference above.

Exemplary Algorithms

In one embodiment, various inventive algorithms are employed to ensure that the quick links that are displayed for the user are intelligently selected for display. Many of the algorithms use dynamically changing information as a basis for ascertaining what quick links to display for the user. Dynamically changing information can include such things as incoming information (e.g. information that is received by the user's computer) and information concerning future activities or events (e.g. calendar appointments). The dynamically changing information is processed by various algorithms to provide the quick links that can be selected for display for the user.

As an example, consider the document-centric single navigable window example above. In that example, the different functionalities include a planner functionality and an email functionality. The planner functionality typically employs information that can be considered as information concerning future activities or events. The email functionality employs information that can be considered as incoming information. The inventive algorithms take into account the nature of this information and attempt to provide an intelligently arranged collection of quick links for the user. Additionally, at least some of the inventive algorithms are employable across different content types. That is, some of the algorithms can provide quick links to different content types. An example of this is given in the "Recent Items List" section below.

The algorithms described below help to determine a set of quick links to provide for users. The inventive algorithms can work in multiple different ways. For example, the algorithms can work:

- As a filter or union of filters on a stored collection or collections of data (e.g. a collection of mail messages or web page favorites); or
- As data tracked in memory about a current application session, usually across multiple types of data (e.g. web pages, calendar appointments, and email messages, for example)

Multiple Selectable Algorithms

In one implementation, a user is presented, via a UI, with multiple algorithms from which they can select to have quick links displayed. The different algorithms can display different collections of quick links depending on the specifics of the algorithm selected by the user. Advantageously, the multiple selectable algorithms can be employed in connection with the single navigable window application described above. Hence, the different selectable algorithms can be employed across different content types.

Consider for example Fig. 8 which shows an exemplary user interface 800 designated as “Favorites”. Interface 800 corresponds to the “Favorites” context block 412 of Fig. 5. Interface 800 includes multiple different algorithms that can be selected by a user. When a user selects a particular algorithm, they are presented with a display of quick links that are provided by that specific selected algorithm. In the illustrated example, four exemplary algorithms are shown: a “Top Favorites” algorithm 802, a “Suggested Favorites” algorithm 804, a “Recently Added Favorites” algorithm 806, a “Places Visited Today” algorithm 808, and a “Recent Items” algorithm 810. The “Top Favorites”, “Suggested Favorites”, and “Recent Items List” algorithms are discussed in specific sections below in more detail. It will be appreciated that the listed algorithms can be provided in any suitable way, e.g. in the illustrated UI or in drop down menus similar to the other quick links.

1 In the single navigable window implementation where a user's context is
2 capable of changing from functionality to functionality, it is important to note that
3 some of the different selectable algorithms, when selected by a user, provide quick
4 links that are particular to the user's present context. That is, as the user's context
5 changes from functionality to functionality, so too do the collection of quick links
6 that are provided by some of the algorithms. For example, if a user is working in
7 their email functionality, then by selecting "Top Favorites", they can see a list of
8 their top favorite email messages. In the present example, their favorite email
9 messages can be displayed directly under interface 800 in a display 812. If a user
10 navigates to the web browser functionality and selects the "Top Favorites"
11 algorithm, they can see a list of their top favorite web sites. Thus, the algorithms
12 are capable of being employed in connection with and across different content
13 types (e.g. email messages and web pages).

14 Fig. 9 is a flow diagram that describes steps in a method in accordance with
15 this described embodiment. The illustrated method can be implemented in any
16 suitable hardware, software, firmware, or combination thereof. In the illustrated
17 example, the method is implemented in software.

18 Step 900 provides multiple different algorithms for displaying quick links.
19 The algorithms that are provided can be any suitable algorithms. Advantageously,
20 some if not all of the algorithms are designed to be employed in connection with
21 and across different content types. In addition, some of the algorithms can display
22 quick links to different content types, as will become apparent below in the
23 "Recent Items List" section. Step 902 displays the multiple different algorithms
24 for selection by a user. The algorithms can be displayed in response to the user
25 actively pulling them up, or they can be displayed automatically when the user's

1 context indicates that the algorithms might be useful to the user. Step 904
2 ascertains whether the user has selected an algorithm. A user can select an
3 algorithm by simply clicking on the appropriate algorithm. If the user has not
4 selected an algorithm, the method can branch back to step 902. Alternately, the
5 method can remove the display of algorithms. If the user selects an algorithm,
6 then step 906 displays quick links that are provided by the algorithm. The quick
7 links can include links that are within the user's present context as well as links
8 that are not within the user's present context.

9 10 **Top Favorites**

11 The inventive Top Favorites algorithm embodiments enable a user to see
12 quick links that are associated with items on a favorites list that have been visited
13 most often by the user as well as items that have most recently been added by the
14 user to a favorites list. To determine which items have been visited "most" often
15 by a user, any suitable metrics can be used. For example, one metric might look at
16 a one-week or a one-month time period and set a predetermined threshold at ten.
17 In this example, an item that is visited more than ten times in the defined time
18 period would be considered as being an item that is visited most often. Similarly,
19 to determine which items have been "most" recently added, any suitable metrics
20 can be used, e.g. added within the past 2 or 3 days. This algorithm recognizes that
21 items of particular interest to a user can include not only those items that a user
22 visits frequently, but items that they recently added to their favorites list as well.

23 The Top Favorites algorithm can be implemented as follows. A database
24 maintains "favorite" entries in which a user has indicated an interest. The
25 database can be maintained in a permanent store. Fig. 10 shows a number of

different exemplary database entries at 1000 that form a portion of such a database. The database entries include a link field 1002 that holds the information describing the link. Here, such information comprises the link's URL. There are also one or more fields 1004 for maintaining information regarding how frequently a user accesses a particular link. In this example, four exemplary fields are provided—each corresponding to a one-week time period. The “1 Wk” field can hold a value associated with a user's access frequency during the preceding week; the “2 Wk” field can hold a value associated with a user's access frequency two weeks ago, and so on. Additionally, a “Date Added” field 1006 includes the date when the user added the link to their favorites list.

In this example, database entries are ranked according to how frequently a user has accessed them. More frequently accessed links are ranked higher than less frequently accessed links. One way of ranking links is to calculate a score for each link that counts the number of times a user has accessed a link, weighting the more recent accesses heavier than the less recent accesses. Each score is then ordered in terms of highest to lowest to provide a ranked list of popular favorites with more popular links appearing toward the top of the list and less popular links appearing toward the bottom of the list or not appear on the list at all. Individual links can now be identified based upon how popular they are as measured by the user's access frequency. Next, the most recently added entries are ascertained in accordance with definable parameters. For example, a search query might specify that entries added within the last two weeks are to be identified. This provides a list of most recently added favorites. This list, and the links from the list of popular favorites are then combined to provide a collection of favorites that includes not only the most popular links (as determined by the user's access

1 frequency), but the most recently added links as well. The latter portion of the list
2 ensures that links that are of current interest to the user populate the “Top
3 Favorites” list.

4 Fig. 11 is a flow diagram that describes steps in a method in accordance
5 with this embodiment. This method can be implemented in any suitable hardware,
6 software, firmware, or combination thereof. In the present example, the method is
7 implemented in software. Step 1100 maintains a database containing information
8 describing various user favorites. This information can include links to the
9 favorites (such as URLs and the like), as well as information that describes the
10 user’s access frequency and when the favorite was added by the user to their
11 favorites list. Exemplary database entries are shown in Fig. 10. Step 1102 runs a
12 first database query that identifies and ranks the most frequently accessed user
13 favorites. Step 1104 runs a second database query that identifies the most recently
14 added favorites. Step 1106 then calculates a union of the first and second queries
15 to provide a user’s “Top Favorites” list.

17 **Suggested Favorites**

18 In another embodiment, a “Suggested Favorites” algorithm enables a user
19 to see links that are associated with items that have been visited most often by the
20 user as well as items that have visited most recently by the user. This algorithm is
21 similar to the “Top Favorites” algorithm, except that instead of running a database
22 query that identifies the most frequently and recently visited favorites (step 1104),
23 a database query is run that identifies the most recently browsed items. Thus, this
24 algorithm recognizes that items of particular interest to a user can include not only
25 those items that a user adds to a manual list and visits frequently, but also items

1 that they most recently visited or visit often and do not add to this list. To
2 determine which items have been most recently visited, any suitable metric can be
3 used, e.g. visited within the past 2 or 3 days.

4 In an exemplary implementation, a “Date Last Accessed” field 1008 (Fig.
5 10) can be included in the database 1000. Entries in this field include the dates
6 when a user last accessed a particular item.

7 Fig. 12 is a flow diagram that describes steps in a method in accordance
8 with this described embodiment. This method can be implemented in any suitable
9 hardware, software, firmware, or combination thereof. In the present example, the
10 method is implemented in software. Step 1200 maintains a database containing
11 information describing various visited pages. This information can include links
12 to the pages (such as URLs and the like), as well as information that describes the
13 user’s access frequency and when the page was last accessed by the user.
14 Exemplary database entries are shown in Fig. 10. Step 1202 runs a first database
15 query that identifies and ranks the most frequently accessed user pages. Step 1204
16 runs a second database query that identifies the most recently accessed pages.
17 Step 1206 then calculates a union of the first and second queries to provide a
18 user’s “Suggested Favorites” list.

20 **Recent Items List**

21 In one embodiment, a “Recent Items List” is provided for a user. This
22 embodiment is particularly useful in the context of the single window application
23 program that provides multiple different functionalities. Recall that each of the
24 different functionalities can have different associated content types, e.g. email
25 messages, calendaring items, contacts, web pages, etc. The “Recent Items List”

1 tracks, in memory, information pertaining to one or more of the last document of a
2 particular content type that the user visited. Advantageously, the application
3 program can store a link to the most recently browsed document of every content
4 type that the application supports. For example, if an application contains email,
5 calendaring, browsing, and contact functionalities, the "Recent Items List" can
6 contain links to each of the most recently accessed documents of the particular
7 content types. This can be extended to include more items in each list or other
8 kinds of extended content: as new functionalities are added to the application, the
9 list could expand to include those types as well. In addition, this algorithm can be
10 extendible to incorporate newly created document types. For example, if a user
11 adds an extension that provides a new document type, this algorithm can ensure
12 that documents of the newly-created document type are included in the "Recent
13 Items List."

14 Consider, for example, Fig. 13 which shows an exemplary "Recent Items
15 List" 1300 that can be maintained in memory. In this example, the list contains
16 five entries, one for each content type that is supported by the application. Each of
17 the entries is a link that is associated with the most recently viewed document of a
18 particular content type. For example, content type 1 might be a link to the last
19 email message that was read, content type 2 might be a link to the last calendaring
20 item that was browsed, content type 3 might be a link to the last web page that was
21 browsed, etc.

22 One particularly useful implementation of the "Recent Items List" occurs in
23 connection with the "back" navigation button drop down menu. Specifically, the
24 navigation bar 408 (see Figs. 5 and 6) includes a "back" navigation button in the
25 form of a leftward-facing arrow. This navigation button includes a drop down

1 menu that can be accessed by clicking on user-engagable indicia 411 located
2 adjacent the button. The drop down menu might list the last three or four items
3 that were most recently encountered by the user. Links to these items are
4 managed in the navigation stack mentioned above. There may be times, however,
5 when a user wishes to access a document on which they previously worked which
6 is not listed in the back drop down menu. For example, consider the following:
7 Assume that a user is browsing through various functionalities and visits an email
8 message from a friend. The email message includes a link that the user follows to
9 a web page. Assume further that the web page includes a 12-page article that the
10 user clicks through. If a user wishes to return to their friend's email message, then
11 they can click the "back" navigation button 12 times to navigate back through the
12 12 pages to get to the email message. Alternately, the user can click the "back"
13 drop down menu to see the last three or four items that they browsed.
14 Accordingly, the user would have to click this drop down menu multiple times.

15 In the described embodiment, the "back" drop down menu supports a
16 "Recent Items List" which contains links to the most recently browsed items of the
17 different content types if they do not appear in the back drop down menu. In the
18 document-centric example above, the "Recent Items List" would contain links to
19 the last mail message, last calendar item, last contact, and last document that the
20 user visited, if those items do not appear in the back drop down menu.

21 Fig. 14 shows an exemplary "back" drop down list 1400 that contains links
22 to three most recently accessed items, as well as a "Recent Items List" that can be
23 clicked on by a user. In the above example where the user has browsed a 12-page
24 document, they would not see a link to their friend's email message in the
25 navigation stack when they pulled down the drop down menu. They would,

1 however, see a link to the email message in the “Recent Items List” because their
2 friend’s email message was the last email message type that was browsed or
3 viewed by the user.

4 5 **Conclusion**

6 The methods and systems described above provide users with a much more
7 user-centric computing experience that is tailored to particular users. The methods
8 and systems provide this user-centric experience while conveniently enabling user
9 participation without requiring the user to change their computing context.
10 Multiple different functionalities can be provided that enable a user to accomplish
11 multiple different tasks. Hence, while a user accomplishes a task associated with
12 one functionality, they can view quick links associated with other functionalities
13 without having to change their present computing context.

14 Although the invention has been described in language specific to structural
15 features and/or methodological steps, it is to be understood that the invention
16 defined in the appended claims is not necessarily limited to the specific features or
17 steps described. Rather, the specific features and steps are disclosed as preferred
18 forms of implementing the claimed invention.

CLAIMS

1. A method of providing information to a computer user comprising:
displaying, in a display area of a user interface, a first contextual display associated with a first context that can enable a user to accomplish one or more tasks, said displaying being accomplished using a single application program that is configured to provide multiple different contexts; and

without changing the first context and using the single application program, presenting quick links to one or more contexts that are different from the user's current context and that are provided by the single application program, each context being associated with a functionality that can enable the user to accomplish various tasks that are different from the one or more tasks that the user can accomplish using the first contextual display.

2. The method of claim 1, wherein each functionality comprises a document-centric functionality.

3. The method of claim 1, wherein each of the functionalities is different.

4. The method of claim 1 further comprising displaying user-engagable indicia, each of which being associated with one or more quick links, said indicia being configured for engagement by a user so that the user can view the associated one or more quick links.

1 5. The method of claim 1, wherein said displaying is accomplished by
2 the single application program using a single window, the application program
3 being configured to navigate the single window between different contexts
4 responsive to the user selecting a quick link.

5
6 6. The method of claim 1 further comprising prior to said presenting,
7 automatically determining at least some of the quick links based, at least in part,
8 on a user's behavior within the single application program.

9
10 7. The method of claim 1 further comprising prior to said presenting,
11 automatically determining at least some of the quick links based, at least in part,
12 on a user's history within the single application program.

13
14 8. The method of claim 1, wherein the single application program is
15 configured with navigation instrumentalities that enable a user to navigate back
16 and forth between the multiple different contexts.

17
18 9. The method of claim 8, wherein the single application program is
19 configured with a navigation model to manage navigation activities of the user, the
20 navigation model comprising a navigation stack.

21
22 10. The method of claim 1 further comprising:
23 presenting the user with a choice of multiple different algorithms, each
24 algorithm being configured to provide a different collection of quick links; and
25

1 said presenting of the quick links being performed responsive to a user
2 selecting one of the multiple different algorithms.

3
4 **11.** The method of claim 10, wherein one of the multiple different
5 algorithms comprises an algorithm that presents quick links on a favorites list
6 based on items visited most often by a user in combination with items that have
7 been recently added by a user to a favorites list.

8
9 **12.** The method of claim 10, wherein one of the multiple different
10 algorithms comprises an algorithm that presents quick links based on items visited
11 most often by a user in combination with items that have been recently visited by a
12 user.

13
14 **13.** The method of claim 10, wherein one of the algorithms comprises
15 an algorithm that presents multiple quick links each of which representing a
16 different document type that was the last item of a particular document type that
17 was visited by a user

18
19 **14.** The method of claim 10, wherein some of the algorithms are
20 employable across multiple different content types.

21
22 **15.** One or more computer-readable media having computer-readable
23 instructions thereon which, when executed by a computer, cause the computer to:

24 provide multiple different functionalities within the confines of a single
25 application program, the multiple different functionalities being associated with

1 individual different document-centric tasks that can be accomplished by a user,
2 individual document-centric tasks being associated with different document types;

3 define a single navigable window within which the different functionalities
4 can be presented to a user so that they can accomplish a task associated with a
5 particular functionality;

6 define individual user-engagable indicia and associate those indicia with
7 one or more of the multiple different functionalities, each indicia being engagable
8 by a user to display quick links that are associated with a functionality, individual
9 quick links being associated with a document that can enable a user to accomplish
10 a task; and

11 display one or more of the quick links associated with one functionality,
12 while a user is engaged in a task associated with another of the functionalities,
13 without requiring the user to change the functionality within which they are
14 working.

15
16 **16.** The computer-readable media of claim 15, wherein the instructions
17 cause the computer to display a selection of multiple different algorithms from
18 which a user can choose and which affect the quick links that are displayed.

19
20 **17.** The computer-readable media of claim 16, wherein one of the
21 multiple different algorithms comprises an algorithm that presents quick links on a
22 favorites list based on items visited most often by a user in combination with items
23 that have been recently added by a user to a favorites list.

1 **18.** The computer-readable media of claim 16, wherein one of the
2 multiple different algorithms comprises an algorithm that presents quick links
3 based on items visited most often by a user in combination with items that have
4 been recently visited by a user.

5
6 **19.** The computer-readable media of claim 16, wherein one of the
7 algorithms comprises an algorithm that presents multiple quick links each of
8 which representing a different document type that was the last item of a particular
9 document type that was visited by a user.

10
11 **20.** The computer-readable media of claim 19, wherein said algorithm
12 that presents multiple quick links is extendable to include newly created document
13 types.

14
15 **21.** The computer-readable media of claim 16, wherein the multiple
16 different algorithms comprise one or more of the following:

17 an algorithm that presents quick links based on items visited most often by
18 a user in combination with items that have been recently added by a user to a
19 favorites list;

20 an algorithm that presents quick links based on items visited most often by
21 a user in combination with items that have been recently visited by a user; and

22 an algorithm that presents multiple quick links each of which representing a
23 different document type that was the last item of a particular document type that
24 was visited by a user.
25

1 **22.** A method of providing information to a computer user comprising:
2 displaying a first contextual display that enables a user to accomplish a task
3 relating to a first content type;
4 displaying quick links associated with one or more content types that are
5 different from the first content type; and
6 responsive to a user selecting a particular quick link, navigating to a content
7 type that is associated with the selected quick link to enable a user to accomplish a
8 different task.

9
10 **23.** The method of claim 22, wherein all of the content types are
11 provided by a single application program.

12
13 **24.** The method of claim 22, wherein all of the content types are
14 provided by a single application program and are displayable within a single
15 navigable window that can be navigated between the content types.

16
17 **25.** The method of claim 22 further comprising prior to displaying said
18 quick links, building said quick links based on dynamically-changing information.

19
20 **26.** The method of claim 22 further comprising prior to displaying said
21 quick links, building said quick links based on dynamically-changing information
22 at least some of which is not related to any actions that the user is taking.

1 **27.** The method of claim 22, wherein said displaying of the quick links
2 comprises doing so using at least one algorithm that can be deployed across
3 multiple different content types.

4
5 **28.** The method of claim 27, wherein one algorithm comprises an
6 algorithm that presents quick links based on items on a favorites list visited most
7 often by a user in combination with items that have been recently added by a user
8 to a favorites list.

9
10 **29.** The method of claim 27, wherein one algorithm comprises an
11 algorithm that presents quick links based on items visited most often by a user in
12 combination with items that have been recently visited by a user.

13
14 **30.** The method of claim 27, wherein one algorithm comprises an
15 algorithm that presents multiple quick links each of which representing a different
16 content type that was the last item of a particular content type that was visited by a
17 user.

18
19 **31.** One or more computer-readable media having computer-readable
20 instructions thereon which, when executed by a computer, cause the computer to:

21 display a first contextual display that enables a user to accomplish a task
22 relating to a first content type;

23 enable a user to select from multiple different algorithms which affect quick
24 links that are displayed and which enable a user to navigate to other contexts, the
25

1 algorithms being deployable across multiple different content types and
2 comprising one or more of the following:

3 an algorithm that presents quick links based on items on a favorites
4 list visited most often by a user in combination with items that have been recently
5 added by a user to a favorites list;

6 an algorithm that presents quick links based on items visited most
7 often by a user in combination with items that have been recently visited by a user;
8 and

9 an algorithm that can present multiple quick links each of which
10 representing a different content type that was the last item of a particular content
11 type that was visited by a user;

12 display quick links associated with one or more content types that are
13 different from the first content type, the quick links being displayed responsive to
14 the user selecting a particular algorithm, all of the content types being provided by
15 a single application program that provides a single navigable window that can be
16 navigated between all of the content types; and

17 responsive to a user selecting a particular quick link, navigate to a content
18 type that is associated with the selected quick link to enable a user to accomplish a
19 different task.

20
21 **32.** A method of providing information to a computer user comprising:
22 receiving information that pertains to multiple different user contexts
23 within an application program;
24
25

1 presenting a display to a user pertaining to a first user context within the
2 application program, the first user context permitting the user to accomplish tasks
3 pertaining to a first content type; and

4 displaying at least one quick link that is associated with a context that is
5 different from the first user context, the displayed quick link being associated with
6 said information and being associated with a different content type.

7
8 **33.** The method of claim 32, wherein the multiple different user
9 contexts are each associated with a different content type.

10
11 **34.** The method of claim 32, wherein said displaying of said at least one
12 quick link comprises displaying multiple quick links, at least some of the quick
13 links being associated with contexts that are each associated with a different
14 content type.

15
16 **35.** The method of claim 32, wherein said displaying comprises
17 displaying the at least one quick link in a drop down menu.

18
19 **36.** The method of claim 32, wherein said displaying comprises doing
20 so without changing content of the display that pertains to the first user context.

21
22 **37.** The method of claim 32, wherein said information comprises
23 information that is generated by the user.

1 **38.** The method of claim 32, wherein said information comprises
2 information that is not generated by the user.

3
4 **39.** The method of claim 32, wherein said information comprises
5 information that can dynamically change.

6
7 **40.** The method of claim 32, wherein said receiving comprises receiving
8 said information while the user is working within the first user context.

9
10 **41.** The method of claim 32 further comprising:
11 receiving user input that selects a displayed quick link; and
12 presenting a display to the user pertaining to a context that is associated
13 with the selected quick link.

14
15 **42.** One or more computers programmed with instructions that cause the
16 computers, when executing the instructions, to:

17 execute an application that is configured to provide multiple different
18 functionalities that can enable a user to accomplish multiple different tasks,
19 individual functionalities being associated with different document types;

20 enable the user to accomplish a task within one of the functionalities and,
21 while doing so, display one or more quick links that are associated with other
22 different functionalities, individual quick links being engagable by the user to
23 navigate to a document type that is associated with that quick link;

24 navigate the user to a item from a particular document type when the user
25 engages a quick link associated with that document type.

1
2 **43.** A computing system comprising:

3 a single application program configured to provide:

4 a single navigable window;

5 multiple different functionalities to which the single navigable
6 window can be navigated by a user; and

7 multiple quick links that are associated with one or more of the
8 multiple different functionalities, individual quick links being displayable
9 and engagable by a user to navigate the single navigable window to the
10 functionalities that are associated with a quick link.

11
12 **44.** The computing system of claim 43, wherein at least some of the
13 different functionalities are associated with different content types.

14
15 **45.** The computing system of claim 43, wherein the single application
16 program is configured to provide multiple different algorithms that are selectable
17 by the user to automatically change quick links that are displayed for them.

18
19 **46.** The computing system of claim 45, wherein at least some of the
20 different algorithms can display links to different content types.

21
22 **47.** The computing system of claim 45, wherein at least some of the
23 different algorithms are configured for use across different content types.

1 **48.** The computing system of claim 47, wherein one of the algorithms
2 comprises an algorithm that presents quick links based on items on a favorites list
3 visited most often by a user in combination with items that have been recently
4 added by a user to a favorites list.

5
6 **49.** The computing system of claim 47, wherein one of the algorithms
7 comprises an algorithm that presents quick links based on items visited most often
8 by a user in combination with items that have been recently visited by a user.

9
10 **50.** The computing system of claim 47, wherein one of the algorithms
11 comprises an algorithm that can present multiple quick links each of which
12 representing a different content type that was the last item of a particular content
13 type that was visited by a user.

14
15 **51.** The computing system of claim 43, wherein the single application
16 program is configured to provide a navigation model that manages the user's
17 navigation activities within the single application program.

18
19 **52.** The computing system of claim 51, wherein the navigation model
20 comprises a back-and-truncate stack.

21
22 **53.** Software code embodied on a computer-readable medium which,
23 when executed by a computer, provides a user interface (UI) comprising:
24 a single window that is capable of being navigated to and between multiple
25 different functionalities that enable a user to accomplish multiple tasks in

1 connection with a single application that provides the multiple different
2 functionalities;

3 links associated with the different functionalities and configured to enable
4 the user to navigate the single window to and between the multiple different
5 functionalities; and

6 user-engagable indicia associated with one or more of the links, the user-
7 engagable indicia being engagable by a user to display quick links that are
8 associated with a particular functionality, the quick links being engagable by the
9 user to automatically navigate the single window to a functionality with which the
10 quick link is associated.

11
12 **54.** The software code of claim 53, wherein the UI further comprises at
13 least one command area that is configured to present context-sensitive commands
14 that automatically change as the user's context changes when they navigate to and
15 between the multiple different functionalities.

16
17 **55.** The software code of claim 54, wherein said at least one command
18 area is configured to display a context block that contains multiple algorithms
19 from which a user can select to vary a list of quick links that are displayed for the
20 user.

21
22 **56.** The software code of claim 55, wherein at least some of the
23 algorithms are employable with different content types.

1 **57.** The software code of claim 53, wherein the UI further comprises
2 browser-like navigation buttons that are engagable by the user for navigating to
3 and between the multiple different functionalities.

4
5 **58.** A computer embodying the computer-readable medium of claim 53.
6

7 **59.** A method of displaying quick links to user information comprising:
8 displaying multiple different algorithms from which a user can select, the
9 algorithms being configured to display quick links to which a user can navigate,
10 individual algorithms being employable across different content types;
11 receiving a user selection of an individual algorithm; and
12 responsive to receiving the user selection, displaying one or more quick
13 links that are provided by the selected algorithm.
14

15 **60.** The method of claim 59, wherein individual algorithms are
16 configured to process dynamically changing information to determine which quick
17 links to display.
18

19 **61.** The method of claim 59, wherein said acts of displaying the multiple
20 different algorithms, receiving the user selection, and displaying the one or more
21 quick links are performed by a single application program that is configured to
22 provide multiple different functionalities that can enable a user to accomplish
23 multiple different tasks, individual quick links being associated with individual
24 functionalities.
25

1 **62.** The method of claim 61, wherein the single application program is
2 configured to provide a single navigable window that can be navigated to and
3 between the multiple different functionalities.

4
5 **63.** The method of claim 59, wherein one of the algorithms is a top
6 favorites algorithm that enables the user to view quick links associated with items
7 on a favorites list that have been visited most often by the user as well as items
8 that have been most recently added to a user's favorites list.

9
10
11 **64.** The method of claim 59, wherein one of the algorithms is a
12 suggested favorites algorithm that enables the user to view quick links associated
13 with items that have been visited most often by the user as well as items that have
14 been most recently visited by the user.

15
16 **65.** The method of claim 59, wherein one of the algorithms is a recent
17 items list that is configured to display multiple items, each of which comprising a
18 different content type that was the last item of a particular content type that was
19 visited by a user.

20
21 **66.** The method of claim 59, wherein the algorithms comprise one or
22 more of:

23 a top favorites algorithm that enables the user to view quick links
24 associated with items on a favorites list that have been visited most often by the
25 user as well as items that have been most recently added to a user's favorites list;

1 a suggested favorites algorithm that enables the user to view quick links
2 associated with items that have been visited most often by the user as well as items
3 that have been most recently visited by the user; and

4 a recent items list that is configured to display multiple items, each of
5 which comprising a different content type that was the last item of a particular
6 content type that was visited by a user.

7
8 **67.** One or more computer-readable media having computer-readable
9 instructions thereon which, when executed by a computer, cause the computer
10 to implement the method of claim 59.

11
12 **68.** One or more computer-readable media having computer-readable
13 instructions thereon which, when executed by a computer, cause the computer to:
14 display multiple different algorithms from which a user can select, the
15 algorithms being configured to display quick links to which a user can navigate,
16 individual algorithms being employable across different content types and
17 comprising one or more of the following:

18 a top favorites algorithm that enables the user to view quick links
19 associated with items that have been visited most often by the user as well
20 as items that have been most recently added to a user's favorites list;

21 a suggested favorites algorithm that enables the user to view quick
22 links associated with items that have been visited most often by the user as
23 well as items that have been most recently visited by the user; and
24
25

1 a recent items list that is configured to display multiple items, each
2 of which comprising a different content type that was the last item of a
3 particular content type that was visited by a user;
4 receive a user selection of an individual algorithm; and
5 responsive to receiving the user selection, display one or more quick links
6 that are provided by the selected algorithm.

7
8 **69.** A method of ascertaining quick links to user information
9 comprising:

10 providing a ranked list of favorite links, the list being based upon the
11 recency and frequency with which a user accesses items associated with the links;

12 providing a most recently added favorites list of links that is based upon
13 when particular links were added to a user's favorites list; and

14 calculating a union of the lists to provide a collection of favorite quick
15 links.

16
17 **70.** The method of claim 69, wherein the ranked list of favorite links is
18 provided by calculating a weighted score for each link that considers the number
19 of times a user accessed an item associated with the link, the score weighting more
20 recent link accesses heavier than less recent link accesses.

21
22 **71.** One or more computer-readable media having computer-readable
23 instructions thereon which, when executed by a computer, implement the method
24 of claim 69.

1 **72.** A method of ascertaining quick links to user information
2 comprising:

3 maintaining a database that contains information describing various user
4 favorite links, the information containing data pertaining to a user's link access
5 frequency and when the links were added to the database;

6 running a first database query that identifies and ranks most frequently
7 accessed links;

8 running a second database query that identifies most recently added
9 favorites; and

10 calculating a union of the first and second database queries to provide quick
11 links comprising a top favorites list.

12
13 **73.** The method of claim 72 further comprising running the first and
14 second queries across multiple content types.

15
16 **74.** One or more computer-readable media having computer-readable
17 instructions thereon which, when executed by a computer, implement the method
18 of claim 72.

19
20 **75.** A method of ascertaining quick links to user information
21 comprising:

22 providing a ranked list of favorite links, the list being based upon the
23 frequency with which a user accesses items associated with the links;

24 providing a most recently accessed favorites list of links that is based upon
25 when particular links were last accessed by a user; and

1 calculating a union of the lists to provide a collection of favorite quick
2 links.

3
4 **76.** The method of claim 75, wherein the ranked list of favorite links is
5 provided by calculating a weighted score for each link that considers the number
6 of times a user accessed an item associated with the link, the score weighting more
7 recent link accesses heavier than less recent link accesses.

8
9 **77.** One or more computer-readable media having computer-readable
10 instructions thereon which, when executed by a computer, implement the method
11 of claim 75.

12
13 **78.** A method of ascertaining quick links to user information
14 comprising:

15 maintaining a database that contains information describing various user
16 visited links, the information containing data pertaining to a user's link access
17 frequency and when the links were last accessed by the user;

18 running a first database query that identifies and ranks most frequently
19 accessed links;

20 running a second database query that identifies most recently accessed
21 links; and

22 calculating a union of the first and second database queries to provide quick
23 links comprising a suggested favorites list.

1 **79.** The method of claim 78 further comprising running the first and
2 second queries across multiple content types.

3
4 **80.** One or more computer-readable media having computer-readable
5 instructions thereon which, when executed by a computer, implement the method
6 of claim 78.

7
8 **81.** A method of ascertaining quick links to user information
9 comprising:

10 maintaining, for multiple different content types, information pertaining to
11 one or more of a last document of a particular content type that was accessed by a
12 user;

13 storing a link for the one or more last documents; and

14 displaying a list for a user comprising the stored links.

15
16 **82.** The method of claim 81, wherein said acts of maintaining, storing,
17 and displaying are performed by a single application program that supports the
18 multiple different content types.

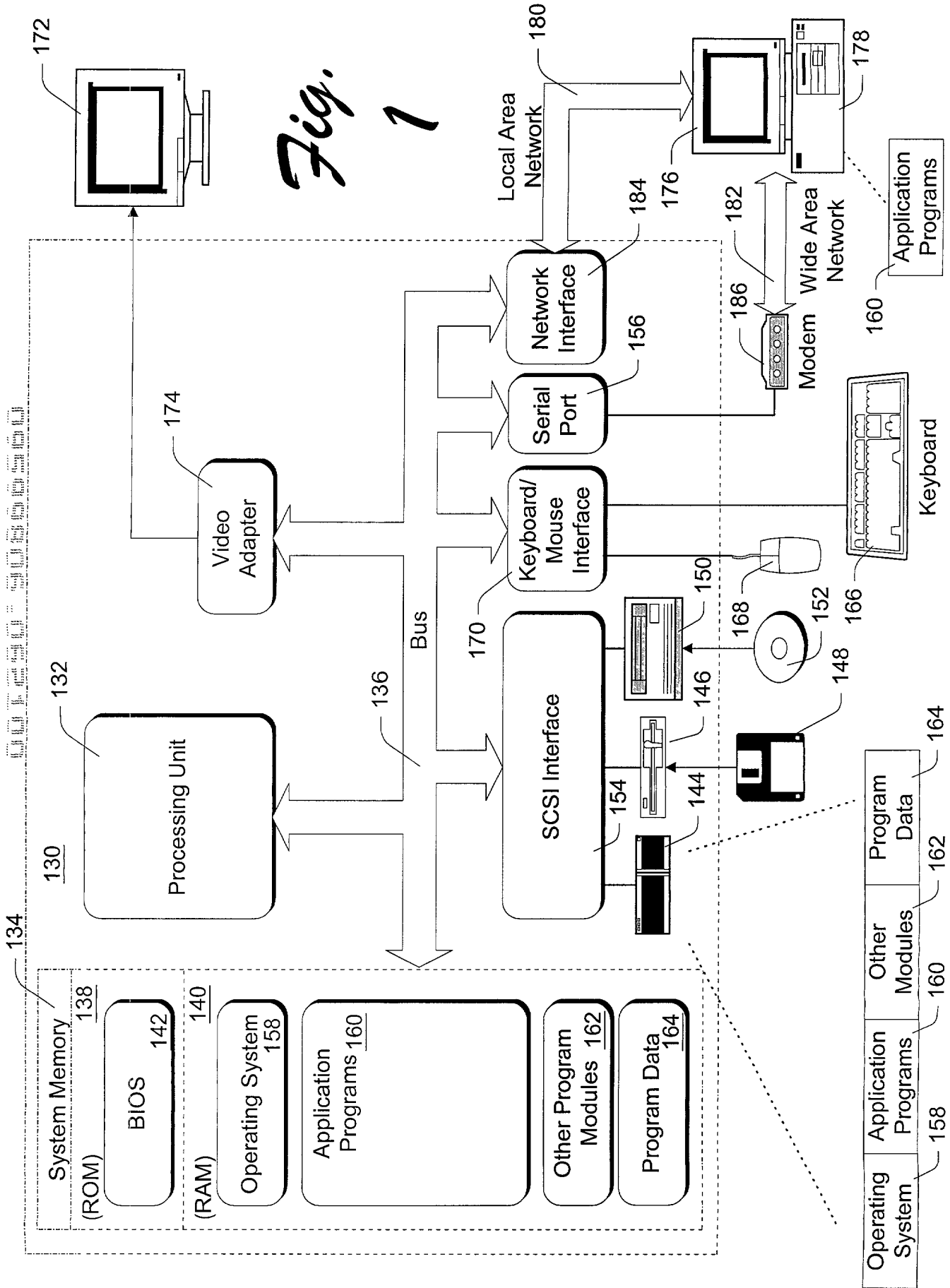
19
20 **83.** The method of claim 81 further comprising, prior to said displaying,
21 ascertaining whether any of the links are already displayable in connection with a
22 drop down menu for a navigation button and, if so, not displaying those links in
23 the list.

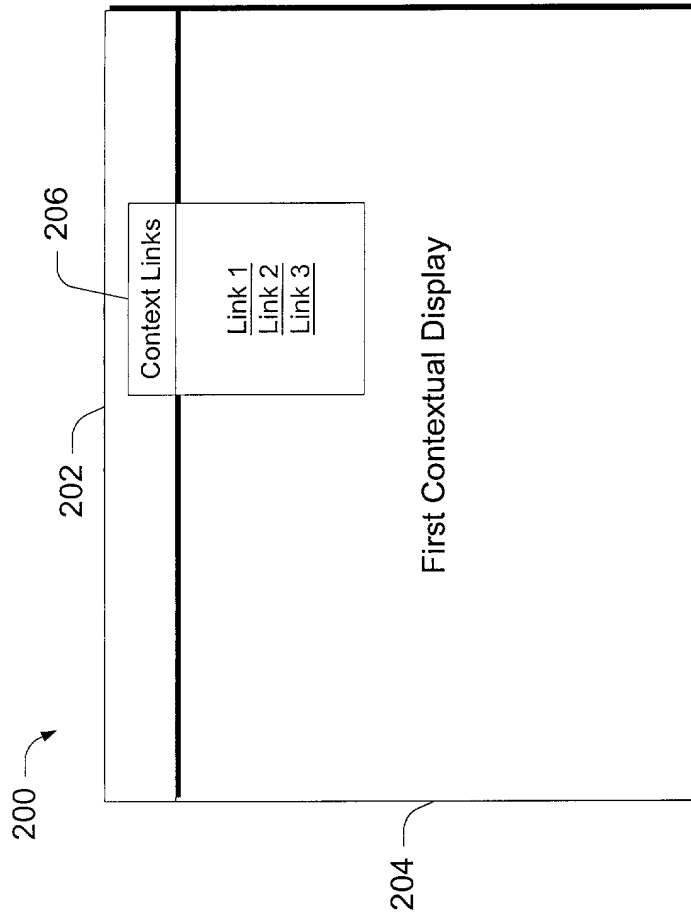
1 **84.** One or more computer-readable media having computer-readable
2 instructions thereon which, when executed by a computer, implement the method
3 of claim 81.
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

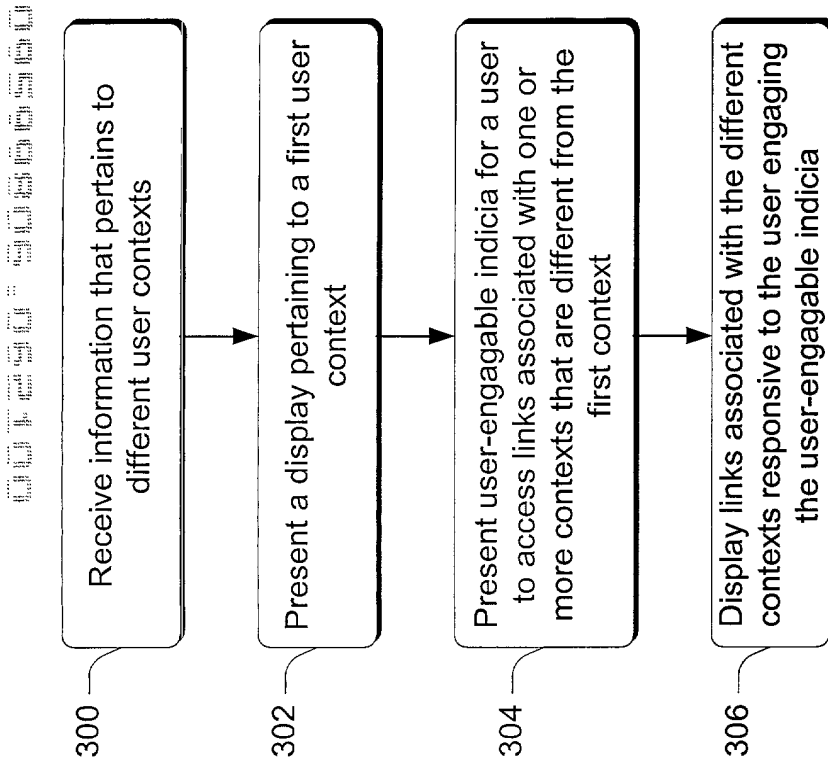
ABSTRACT

Methods and systems of providing information to computer users are described. In one embodiment, a user interface is provided having a display area that permits a user to accomplish various tasks. Individual tasks can be associated with individual different functionalities which can enable the user to accomplish tasks in different user contexts, e.g. word processing tasks, email tasks, web browsing tasks and the like. The different tasks can advantageously pertain to different content types or document types. When a user is working within a particular context, they can view quick links that are associated with one or more of the different contexts without having to change their particular context. By clicking on a quick link, the user then can automatically navigate to the context associated with the quick link so that they can then accomplish context-specific tasks. Advantageously, the various functionalities or contexts can be provided by a single application program that also manages the user's navigation activities to and between the different functionalities. In various embodiments, a user can select from among a number of different algorithms that are deployable across different content types and which affect which quick links are displayed.

Fig. 1



*Fig. 2*

*Fig. 3*

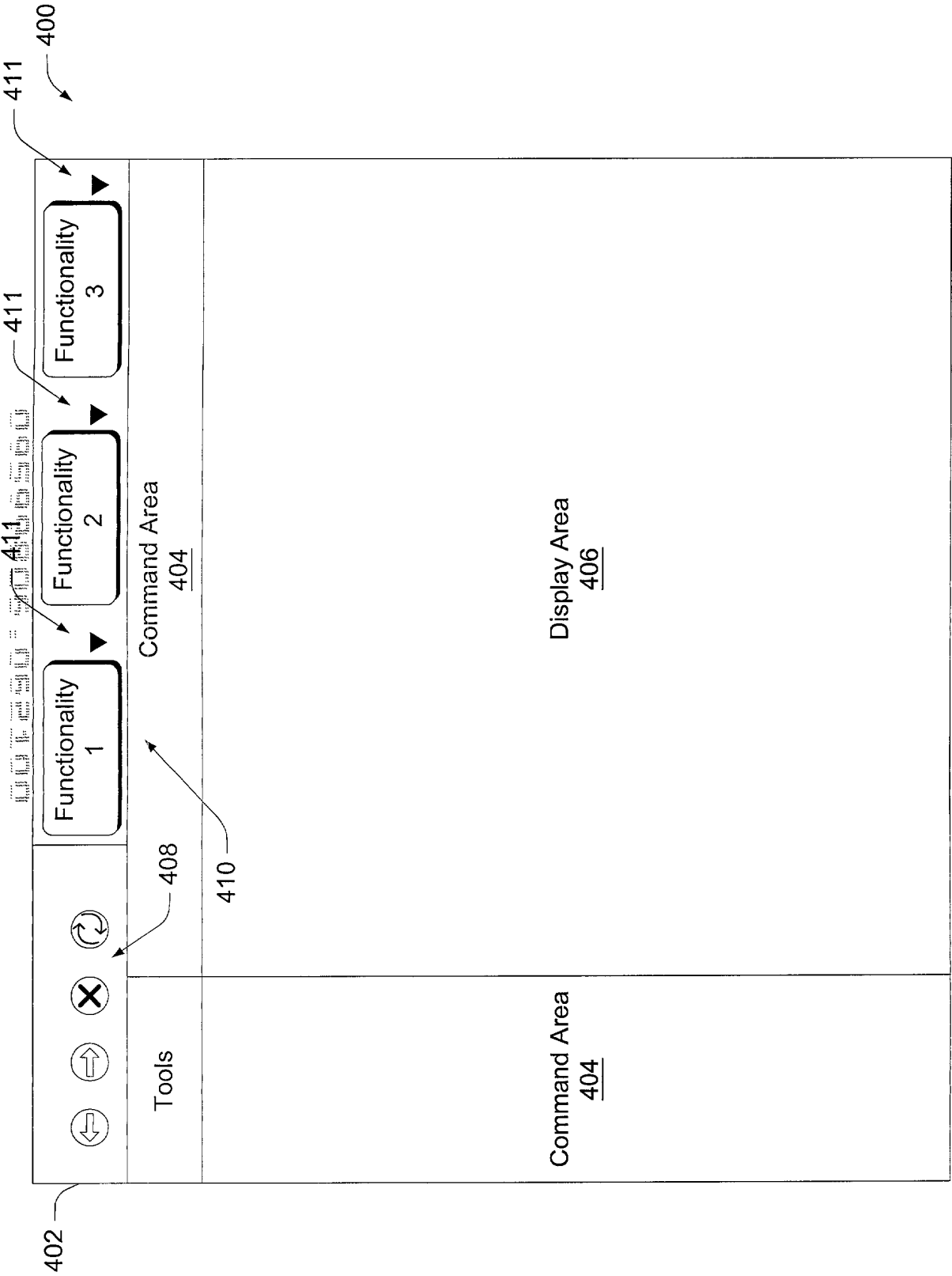


Fig. 4

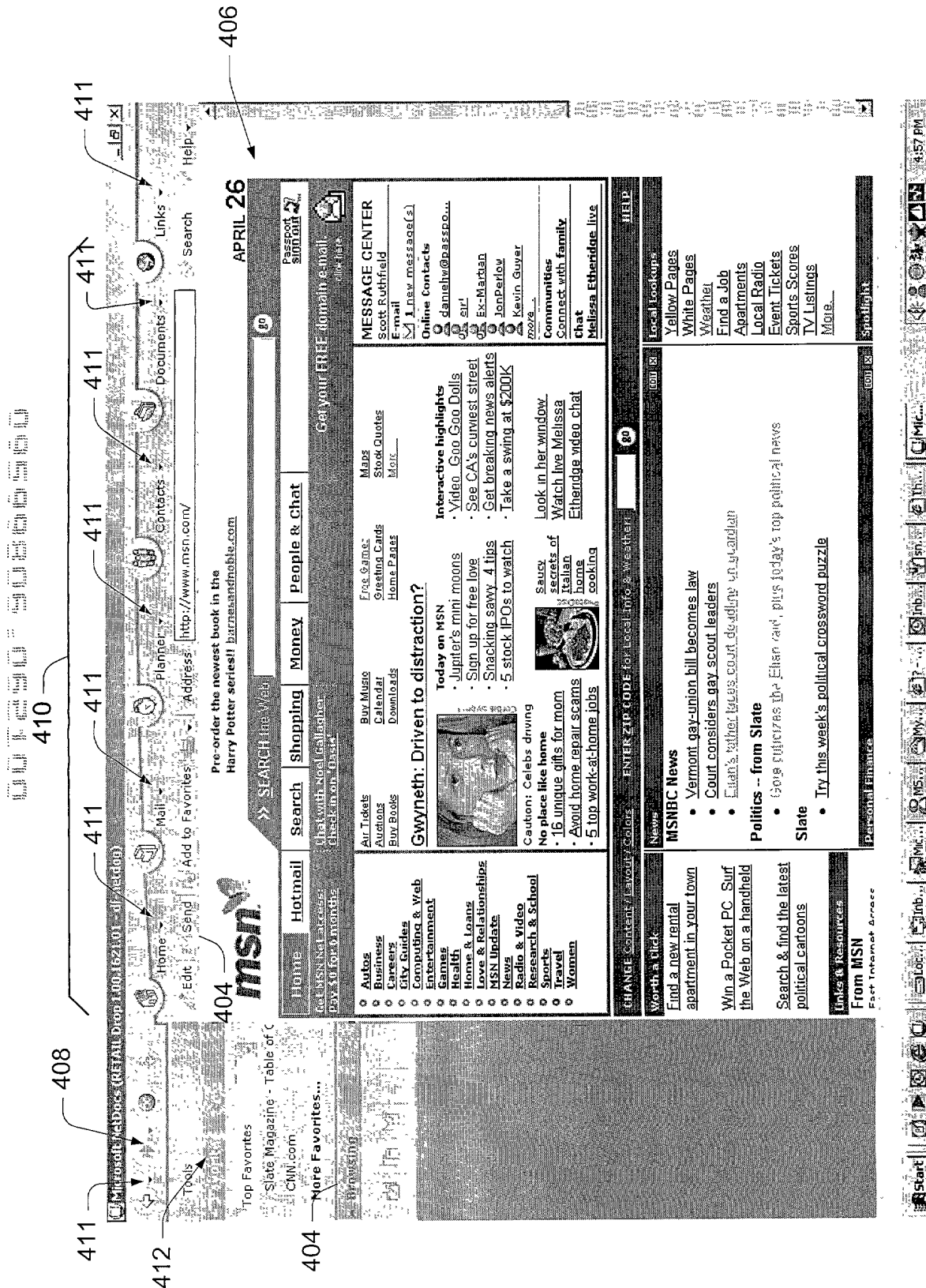
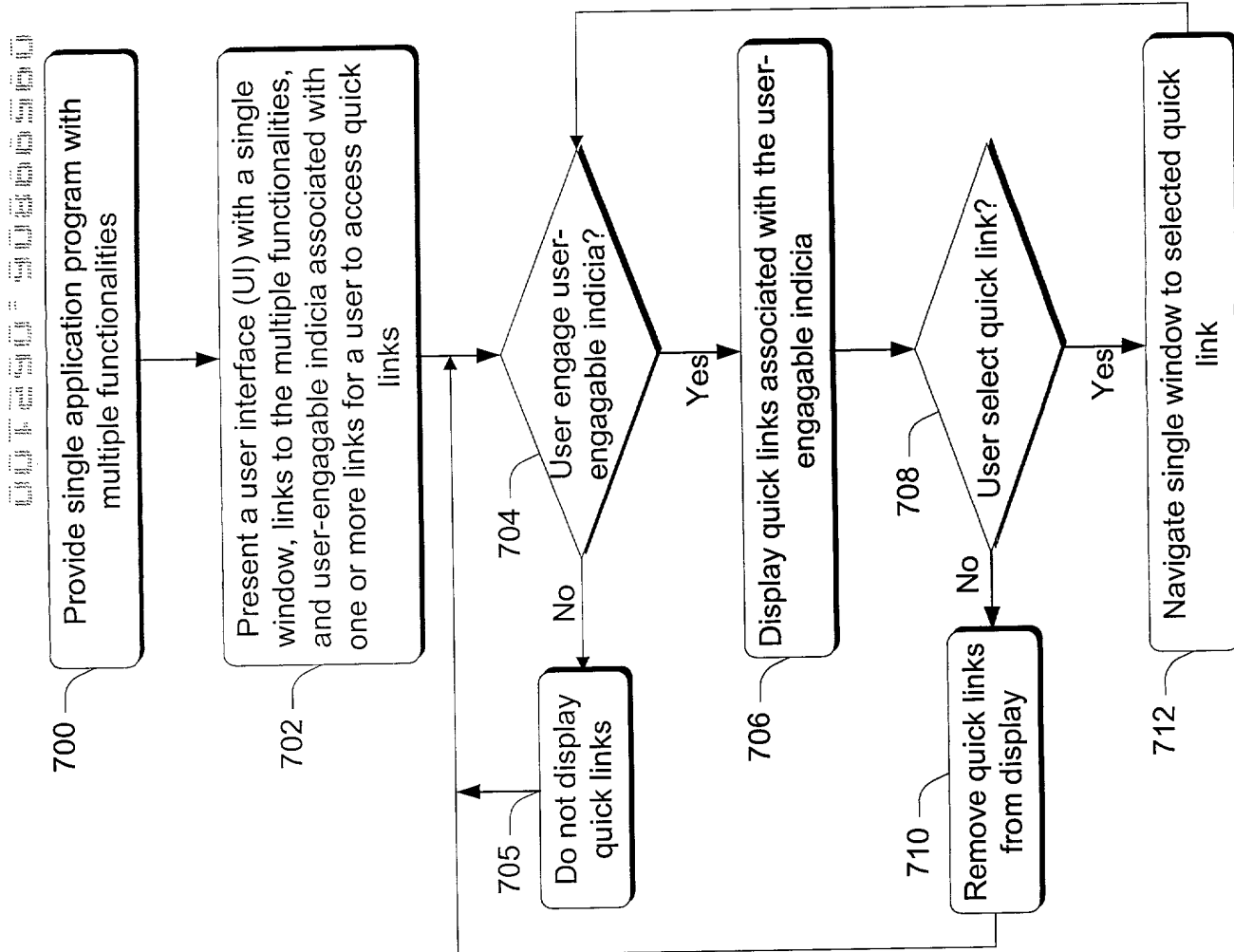


Fig. 5



Fig. 7



900 902 904 906

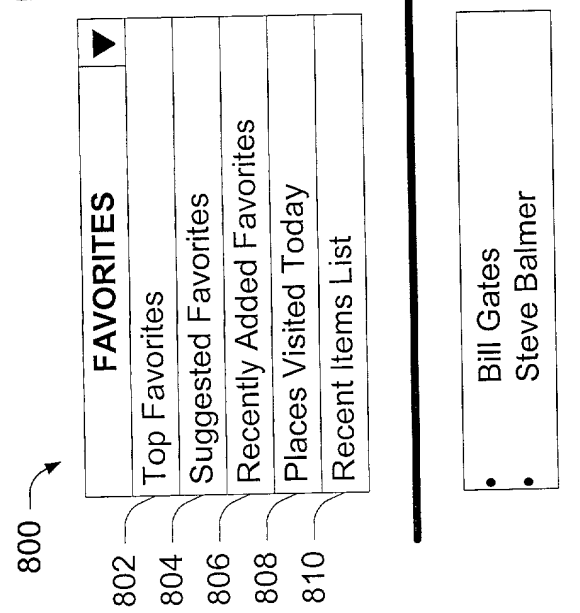


Fig. 8

800

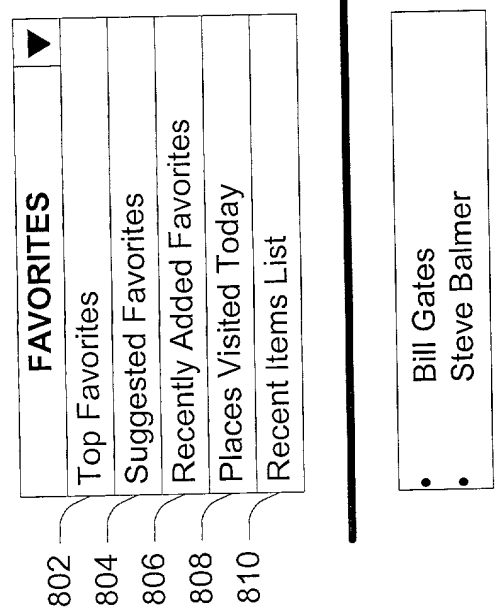


Fig. 9

1000 →

1002

1004

1006

1008

LINKS	1 Wk	2 Wk	3 Wk	4 Wk	Date Added	Date Last Accessed
www.msn.com	10	2	4	5	4/1/00	4/10/00
www.abc.com	7	8	4	1	3/2/00	4/20/00
www.xyz.com	10	8	7	2	8/1/99	5/20/00
www.cbs.com	1	4	8	10	11/9/99	4/20/00
www.espn.com	10	1	1	1	5/2/99	5/5/00

Fig. 10

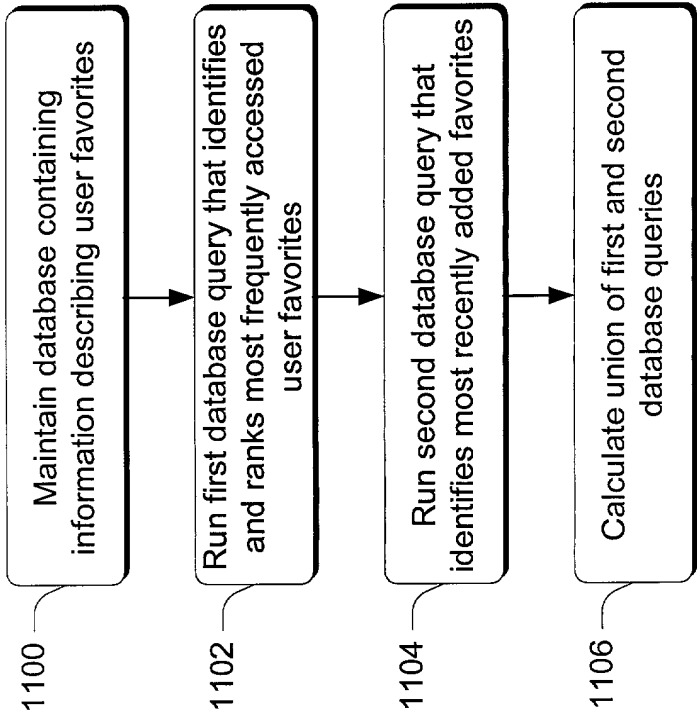


Fig. 11

Fig. 12

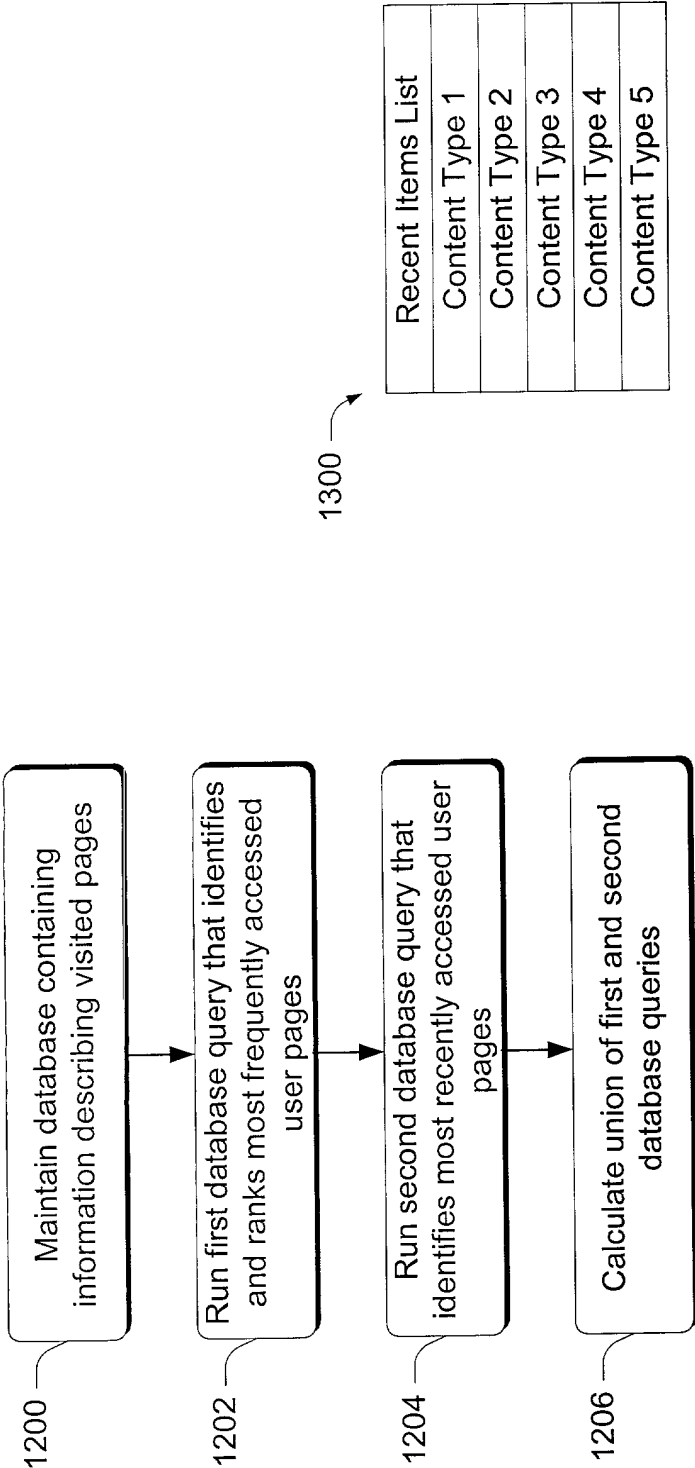
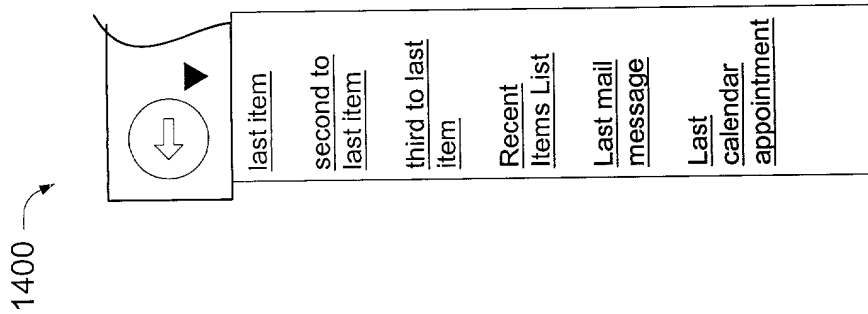


Fig. 13

Fig. 12

*Fig. 14*